

Bounded Turing Reductions and Data Processing Inequalities for Sequences *

Adam Case
 Department of Computer Science
 Iowa State University
 Ames, IA 50011 USA

Abstract

A *data processing inequality* states that the quantity of shared information between two entities (e.g. signals, strings) cannot be significantly increased when one of the entities is processed by certain kinds of transformations. In this paper, we prove several data processing inequalities for sequences, where the transformations are bounded Turing functionals and the shared information is measured by the lower and upper mutual dimensions between sequences.

We show that, for all sequences X, Y , and Z , if Z is computable Lipschitz reducible to X , then

$$mdim(Z : Y) \leq mdim(X : Y) \text{ and } Mdim(Z : Y) \leq Mdim(X : Y).$$

We also show how to derive different data processing inequalities by making adjustments to the computable bounds of the *use* of a Turing functional.

The *yield* of a Turing functional Φ^S with access to at most n bits of the oracle S is the smallest input $m \in \mathbb{N}$ such that $\Phi^{S \upharpoonright n}(m) \uparrow$. We show how to derive *reverse data processing inequalities* (i.e., data processing inequalities where the transformation may significantly *increase* the shared information between two entities) for sequences by applying computable bounds to the yield of a Turing functional.

1 Introduction

Various branches of information theory have developed methods for measuring the shared information between two objects. It is expected that a measure of mutual information satisfy certain properties (e.g., see [2]). Perhaps the most important property is the *data processing inequality*, which says that the quantity of shared information between two objects cannot be significantly increased when one of the objects is processed by certain transformations [5].

In algorithmic information theory, if $f : \Sigma^* \rightarrow \Sigma^*$ is a partial computable function, then there is a constant $c \in \mathbb{N}$ such that, for all strings $x, y \in \Sigma^*$,

$$I(f(x) : y) \leq I(x : y) + c, \tag{1.1}$$

*This research was supported in part by National Science Foundation Grants 1247051 and 1545028. A preliminary version of part of this work was presented at the 11th International Conference on Computability, Complexity, and Randomness.

where $I(x : y) = K(y) - K(y | x)$ is the *algorithmic mutual information* between strings x and y [10]. While (1.1) is a data processing inequality for strings, there still exist settings within algorithmic information theory that do not have known data processing inequalities.

In this paper, we discuss several new data processing inequalities for sequences. We use *mutual dimension*, a recent development in *constructive dimension*, as the means for measuring the quantity of shared information between two sequences [3, 4]. Lutz defined and explored the constructive dimension of sequences in [11], and Mayordomo showed that constructive dimension can be characterized in terms of Kolmogorov complexity in [12]. Mutual dimension is a generalization of constructive dimension and is defined in terms of algorithmic mutual information. Formally, the *lower* and *upper* mutual dimensions between sequences $S \in \Sigma^\infty$ and $T \in \Sigma^\infty$ are defined by

$$mdim(S : T) = \liminf_{n \rightarrow \infty} \frac{I(S \upharpoonright n : T \upharpoonright n)}{n \log |\Sigma|}$$

and

$$Mdim(S : T) = \limsup_{n \rightarrow \infty} \frac{I(S \upharpoonright n : T \upharpoonright n)}{n \log |\Sigma|},$$

respectively. Intuitively, these are the lower and upper *densities* of algorithmic mutual information between S and T . Originally, Case and Lutz defined the lower and upper mutual dimensions between points in Euclidean space and showed that they have *all* of the expected properties that a measure of mutual information should have, including a data processing inequality [3]. In a recent follow-up paper, the same authors extend this notion of mutual dimension to sequences and proved that it has several desirable properties [4]. However, no discussion regarding data processing inequalities for sequences was provided. Our primary goal in the present paper is to analyze how the lower and upper mutual dimensions between two sequences change when one of the sequences is transformed by a Turing functional.

A reduction can be described in several ways. Generally speaking, a problem A reduces to a problem B if A is solvable when assuming that B is solvable. In computability theory, Turing reductions are used to discuss the idea of relative computability. Formally, a sequence S is *Turing reducible* to a sequence T if there exists an oracle machine that computes S when T is written on the oracle tape. We often refer to oracle machines as Turing functionals, which have been studied in detail by Rogers [13] and Soare [14, 15]. When a Turing functional Φ^S runs on a particular input, it is allowed to query the oracle S at any time. The *use* of a Turing functional is the largest position of the oracle tape that is queried during the computation of Φ^S on input n . We will be primarily concerned with Turing functionals whose use is bounded by a computable function.

Downey, Hirshfeldt, and LaForte first defined *sw-reducibility* (strong weak truth table reducibility) as a Turing reduction whose use is bounded by $n + c$, where $n \in \mathbb{N}$ is the input and c is a constant [6]. The authors showed that, for all sequences S and T , if T is sw-reducible to S , then, for all $n \in \mathbb{N}$,

$$K(T \upharpoonright n) \leq K(S \upharpoonright n) + O(1).$$

A sw-reduction is now referred to as a *computable Lipschitz reduction* (*cl-reduction*) because all Turing functionals whose use is bounded by $n + c$ can be viewed as an effective Lipschitz continuous function [9, 8].

In section 3, we discuss data processing inequalities for sequences, where transformations are represented by Turing functionals with bounded use. Our main result of this section says that, for

all sequences $X, Y, Z \in \Sigma^\infty$, if Z is cl-reducible to X , then

$$mdim(Z : Y) \leq mdim(X : Y)$$

and

$$Mdim(Z : Y) \leq Mdim(X : Y).$$

We also show that, for all $\alpha \geq 1$, if Z is reducible to X via a functional Φ whose use is bounded by $\lceil \alpha(n + c) \rceil$, for all inputs $n \in \mathbb{N}$, then

$$mdim(Z : Y) \leq \alpha \cdot mdim(X : Y)$$

and

$$Mdim(Z : Y) \leq \alpha \cdot Mdim(X : Y).$$

We then provide weaker versions of the above inequalities stated in terms of the Turing functionals themselves.

In section 4, we explore *reverse data processing inequalities* for sequences, i.e., data processing inequalities where the transformation may significantly *increase* the amount of shared information between two objects. Unlike the data processing inequalities described above, we cannot derive reverse data processing inequalities by restricting how much of the oracle a Turing functional accesses. Instead, we place restrictions on the lengths of the strings that a Turing functional outputs.

In [7], Gács analyzed the lengths of the outputs of monotonic operators, which are also used to describe Turing reductions. Similarly, we are interested in examining the lengths of the strings output by a Turing functional equipped with a finite oracle. We define the *yield* of a Turing functional Φ^S with access to at most $n \in \mathbb{N}$ bits of the oracle S , denoted $\phi_{yield}^S(n)$, to be the smallest input $m \in \mathbb{N}$ such that $\Phi^{S \upharpoonright n}(m) \uparrow$.

We say that a sequence T is *uniquely yield bounded reducible* (*uyb-reducible*) to a sequence S if there exists a Turing functional Φ such that,

1. if the first $\phi_{yield}^S(n)$ symbols of Φ^S is a prefix of Φ^T , then the first n symbols of S is a prefix of T , and
2. $\phi_{yield}^S(n)$ is bounded by a computable function.

Our main result of this section says that, for all sequences $X, Y, Z \in \Sigma^\infty$, if Z is uyb-reducible to X via a functional Φ such that $\phi_{yield}^X(n) \leq n + c$, for some constant $c \in \mathbb{N}$, then

$$mdim(X : Y) \leq mdim(Z : Y)$$

and

$$Mdim(X : Y) \leq Mdim(Z : Y).$$

We also show that, for all $\alpha \geq 1$, if Z is uyb-reducible to X via a functional Φ such that $\phi_{yield}^X(n) \leq \lceil \alpha(n + c) \rceil$, for all inputs $n \in \mathbb{N}$, then

$$mdim(X : Y) \leq \alpha \cdot mdim(Z : Y)$$

and

$$Mdim(X : Y) \leq \alpha \cdot Mdim(Z : Y).$$

2 Preliminaries

We begin by discussing several formal definitions and concepts related to Turing reductions, Kolmogorov complexity, and constructive dimension. Let $\mathbb{N} = \{0, 1, 2, \dots\}$, $\Sigma = \{0, 1, \dots, k-1\}$ be the *alphabet* consisting of k symbols, and Σ^* be the set of all strings over Σ . We write Σ^∞ for the set of all infinite sequences over Σ , and, for every $S \in \Sigma^\infty$ and $n \in \mathbb{N}$, $S[n]$ is the n th symbol of S and $S \upharpoonright n$ denotes the first n symbols of S . For all strings $x, y \in \Sigma^*$ and sequences $S \in \Sigma^\infty$, we write $x \sqsubseteq S$ and $x \sqsubseteq y$ to mean that x is a *prefix* of S and x is a prefix of y , respectively.

Oracle machines are used as a means of carrying out *relative* computations, i.e., computations performed by Turing machines with access to an additional source of information provided by the oracle. An oracle machine is a Turing machine equipped with an additional read-only tape called the *oracle tape*. We write M^S to denote an oracle machine with sequence S written on its oracle tape. Given an input $n \in \mathbb{N}$, an oracle machine will either halt or run forever. If the oracle machine halts on a given input, then it must query the oracle tape a finite number of times.

It is often useful to provide an oracle tape with a string rather than a sequence. The behavior of a machine M with a string oracle $x \in \Sigma^*$ is identical to that of a sequence oracle $S \in \Sigma^\infty$, except that if the machine attempts to query a position of the oracle tape that is larger than $|x| - 1$, the machine immediately enters a looping state and runs forever.

The following notations and definitions can be found in [1, 13, 15]. We may disassociate an oracle machine M from any particular oracle and refer to it as a partial function $\Phi_M : \Sigma^\infty \times \mathbb{N} \rightarrow \Sigma^*$ defined by $\Phi_M(S, n) = M^S(n)$. Each Φ_M is called a *Turing functional*. The partial function $\Phi_M^S : \mathbb{N} \rightarrow \Sigma^*$ is defined by $\Phi_M^S(n) = \Phi_M(S, n)$, and we write $\Phi_M^S(n) \downarrow$ if M^S halts on input n and $\Phi_M^S(n) \uparrow$ if M^S does not halt on input n .

For any two sequences S and T and any oracle machine M , we write $\Phi_M^S = T$ if, for all $n \in \mathbb{N}$,

$$\Phi_M^S(n) = T \upharpoonright n.$$

We say that T is *Turing reducible to S* if there exists an oracle machine M such that $\Phi_M^S = T$.

For the rest of this paper, we omit the M in Φ_M and Φ_M^S and denote an arbitrary Turing functional by Φ and an arbitrary Turing functional with oracle S by Φ^S .

We now provide a brief overview of the basics of *Kolmogorov complexity*. Specifically, we are interested in *prefix-free Kolmogorov complexity*. Therefore, all Turing machines used in the following definitions will be self-delimiting.

Let M be an arbitrary Turing machine. The *conditional Kolmogorov complexity* of $x \in \Sigma^*$ given $y \in \Sigma^*$ with respect to M is

$$K_M(x | y) = \min\{|\pi| \mid \pi \in \{0, 1\}^* \text{ and } M(\pi, y) = x\}.$$

The *Kolmogorov complexity* of $x \in \Sigma^*$ with respect to M is $K_M(x) = K_M(x | \lambda)$, where λ is the *empty string*. We say that a Turing machine M' is *optimal* if, for every Turing machine M , there is a constant $c_M \in \mathbb{N}$ such that, for all $x \in \Sigma^*$,

$$K_{M'}(x) \leq K_M(x) + c_M,$$

where c_M is called an *optimality constant* of M . An important fact in algorithmic information theory is that every universal Turing machine is optimal [10]. Therefore, we fix a particular universal Turing machine U that we reference for the entirety of this paper and define the *Kolmogorov*

complexity of $x \in \Sigma^*$ by $K(x) = K_U(x)$ and the *conditional Kolmogorov complexity* of x given y by $K(x|y) = K_U(x|y)$.

We define the *joint Kolmogorov complexity* of $x \in \Sigma^*$ and $y \in \Sigma^*$ by $K(x, y) = K(\langle x, y \rangle)$, where $\langle \cdot \rangle$ is a string pairing function. The *mutual information* between strings x and y is

$$I(x : y) = K(y) - K(y|x),$$

which is the quantity of algorithmic information that x and y share. For a more thorough discussion on this topic, see [10].

3 Turing Functionals with Bounded Use and Data Processing Inequalities

In this section, we develop data processing inequalities for sequences and show how these inequalities change when applying different computable bounds to the *use* of a Turing functional. First, we prove several supporting lemmas.

Lemma 3.1. *There exists a constant $c \in \mathbb{N}$ such that, for all $u, v, w \in \Sigma^*$,*

$$K(u|vw) \leq K(u|v) + K(|v|) + c.$$

Proof. Let M be a TM such that, if $U(\pi_1) = |v|$ and $U(\pi_2, v) = u$,

$$M(\pi_1\pi_2, vw) = u.$$

Let $c_M \in \mathbb{N}$ be an optimality constant of M . Assume the hypothesis, and let π_1 be a minimum-length program for $|v|$ and π_2 be a minimum-length program for u given v . By optimality,

$$\begin{aligned} K(u|vw) &\leq K_M(u|vw) + c_M \\ &\leq |\pi_1\pi_2| + c_M \\ &= K(u|v) + K(|v|) + c, \end{aligned}$$

where $c = c_M$. □

Corollary 3.2. *For all $u, v, w \in \Sigma^*$,*

$$I(u : w) \leq I(uv : w) + o(|u|).$$

Proof. By the definition of mutual information and Lemma 3.1, there exists a constant $c \in \mathbb{N}$ such that

$$\begin{aligned} I(u : w) &= K(w) - K(w|u) \\ &\leq K(w) - K(w|uv) + K(|u|) + c \\ &= I(uv : w) + o(|u|). \end{aligned} \quad \square$$

The following lemma was proven in [4].

Lemma 3.3. *For all strings $u, w \in \Sigma^*$,*

$$I(u : w) = K(u) + K(w) - K(u, w) + o(|u|).$$

Corollary 3.4. *For all $u, w \in \Sigma^*$,*

$$I(u : w) = I(w : u) + o(|u|) + o(|w|).$$

Proof. By Lemma 3.3,

$$\begin{aligned} I(u : w) &= K(u) + K(w) - K(u, w) + o(|u|) \\ &= K(w) + K(u) - K(w, u) + o(|u|) \\ &= I(w : u) + o(|u|) + o(|w|). \end{aligned}$$

□

The following lemma was proven in [3].

Lemma 3.5. *Let $f : \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$ be a computable function. There exists a constant $c \in \mathbb{N}$ such that, for all strings $u, v, w \in \Sigma^*$,*

$$K(u | w) \leq K(u | f(w, v)) + K(v) + c.$$

We now investigate bounded Turing reductions and their effects on the shared algorithmic information between strings. As previously mentioned, a halting oracle machine computation can only make a finite number of queries to its oracle, and we are often interested in knowing the largest position of the oracle tape that a machine will query before it halts. The following definition is from [1].

Definition. The *use function* of a Turing functional Φ equipped with oracle $S \in \Sigma^\infty$ is

$$\phi_{use}^S(n) = \begin{cases} m + 1 & \text{if } \Phi^S(n) \downarrow \text{ and } m \text{ is the largest query made to the oracle } S \\ 0 & \text{if } \Phi^S(n) \downarrow \text{ and the oracle } S \text{ is not queried during the computation} \\ \text{undefined} & \text{if } \Phi^S(n) \uparrow \end{cases},$$

for every $n \in \mathbb{N}$.

We denote Turing functionals using uppercase Greek letters (e.g., Φ, Γ) and their corresponding use functions by lowercase Greek letters (e.g., ϕ_{use}, γ_{use}).

Definition. A sequence $T \in \Sigma^\infty$ is *bounded Turing reducible* (*bT-reducible*) to a sequence $S \in \Sigma^\infty$ if T is Turing reducible to S by a Turing functional Φ such that ϕ_{use}^S is bounded by a computable function.

For convenience, we say that $T \in \Sigma^\infty$ is *m-bT-reducible* to $S \in \Sigma^\infty$ if T is bT-reducible to S via Φ and $m : \mathbb{N} \rightarrow \mathbb{N}$ is a computable function bounding ϕ_{use}^S .

Lemma 3.6. *Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be an increasing, computable function. For all $X, Y, Z \in \Sigma^\infty$, if Z is m-bT-Turing reducible to X , then*

$$I(Z \upharpoonright n : Y \upharpoonright n) \leq I(X \upharpoonright m(n) : Y \upharpoonright m(n)) + o(m(n)).$$

Proof. Assume that Z is m -bT-Turing reducible to X by some Turing functional Φ whose use function ϕ_{use}^X is bounded by m . By Corollaries 3.2 and 3.4,

$$\begin{aligned} I(Z \upharpoonright n : Y \upharpoonright n) &= I(Y \upharpoonright n : Z \upharpoonright n) + o(n) \\ &\leq I(Y \upharpoonright m(n) : Z \upharpoonright n) + o(n) \\ &= I(Z \upharpoonright n : Y \upharpoonright m(n)) + o(m(n)). \end{aligned} \tag{3.1}$$

Define the partial function $f : \{0, 1\}^* \times \mathbb{N} \rightarrow \{0, 1\}^*$ by

$$f(u, n) = \Phi^u(n),$$

for all $u \in \Sigma^*$ and $n \in \mathbb{N}$. The function f is clearly computable. Therefore, by (3.1) and Lemma 3.5,

$$\begin{aligned} I(Z \upharpoonright n : Y \upharpoonright n) &\leq I(f(X \upharpoonright m(n), n) : Y \upharpoonright m(n)) + o(m(n)) \\ &= K(Y \upharpoonright m(n)) - K(Y \upharpoonright m(n) | f(X \upharpoonright m(n), n)) + o(m(n)) \\ &\leq K(Y \upharpoonright m(n)) - K(Y \upharpoonright m(n) | X \upharpoonright m(n)) + o(m(n)) \\ &= I(X \upharpoonright m(n) : Y \upharpoonright m(n)) + o(m(n)). \end{aligned} \quad \square$$

The first notion of mutual dimension was defined in [3] to analyze the density of algorithmic mutual information between points in Euclidean space. It was then extended to sequences in [4] in order to study coupled randomness.

Definition. The *lower* and *upper mutual dimensions* between $S \in \Sigma^\infty$ and $T \in \Sigma^\infty$ are

$$mdim(S : T) = \liminf_{n \rightarrow \infty} \frac{I(S \upharpoonright n : T \upharpoonright n)}{n \log |\Sigma|}$$

and

$$Mdim(S : T) = \limsup_{n \rightarrow \infty} \frac{I(S \upharpoonright n : T \upharpoonright n)}{n \log |\Sigma|},$$

respectively. We now present an important technical lemma.

Lemma 3.7 (Bounded Use Processing Lemma). *Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be an increasing, computable function. For all $X, Y, Z \in \Sigma^\infty$, if Z is m -bT-Turing reducible to X , then*

$$mdim(Z : Y) \leq mdim(X : Y) \left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right)$$

and

$$Mdim(Z : Y) \leq Mdim(X : Y) \left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right),$$

except when $\left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right) = \infty$ while either $mdim(X : Y) = 0$ or $Mdim(X : Y) = 0$.

Proof. By Lemma 3.6,

$$\begin{aligned}
mdim(Z : Y) &= \liminf_{n \rightarrow \infty} \frac{I(Z \upharpoonright n : Y \upharpoonright n)}{n \log |\Sigma|} \\
&\leq \liminf_{n \rightarrow \infty} \frac{I(X \upharpoonright m(n) : Y \upharpoonright m(n)) + o(m(n))}{n \log |\Sigma|} \\
&= \liminf_{n \rightarrow \infty} \left(\frac{I(X \upharpoonright m(n) : Y \upharpoonright m(n)) + o(m(n))}{m(n) \log |\Sigma|} \cdot \frac{m(n)}{n} \right) \\
&\leq \left(\liminf_{n \rightarrow \infty} \frac{I(X \upharpoonright m(n) : Y \upharpoonright m(n)) + o(m(n))}{m(n) \log |\Sigma|} \right) \left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right) \\
&= mdim(X : Y) \left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right).
\end{aligned}$$

A similar proof can be given for $Mdim$. □

Definition. Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be defined by $m(n) = n + c$, where $c \in \mathbb{N}$ is a constant. A sequence $T \in \Sigma^\infty$ is *computable Lipschitz reducible* (*cl-reducible*) to a sequence $S \in \Sigma^\infty$ if T is m -bT-reducible to S .

The following theorem follows directly from the Bounded Use Processing Lemma.

Theorem 3.8. *For all sequences $X, Y, Z \in \Sigma^\infty$, if Z is cl-reducible to X , then*

$$mdim(Z : Y) \leq mdim(X : Y)$$

and

$$Mdim(Z : Y) \leq Mdim(X : Y).$$

Let $\alpha \geq 1$ and $h_\alpha : \mathbb{N} \rightarrow \mathbb{N}$ be defined by $h_\alpha(n) = \lceil \alpha(n + c) \rceil$, where $c \in \mathbb{N}$ is a constant. The following is a corollary of the Bounded Use Processing Lemma.

Corollary 3.9. *Let $\alpha \geq 1$. For all sequences $X, Y, Z \in \Sigma^\infty$, if Z is h_α -bT-reducible to a sequence X , then*

$$mdim(Z : Y) \leq \alpha \cdot mdim(X : Y)$$

and

$$Mdim(Z : Y) \leq \alpha \cdot Mdim(X : Y).$$

Typically, data processing inequalities are statements about *all* of the defined outputs of a particular transformation. The results above, while strong, are not framed in this manner. To remedy this, we now discuss data processing inequalities in terms of individual bounded Turing functionals.

Definition. Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be a computable function. A *m -bounded Turing functional* (*m -bT-functional*) is a Turing functional such that, for every sequence $S \in \Sigma^\infty$ and every $n \in \mathbb{N}$ where $\Phi^S(n)$ is defined, $\phi_{use}^S(n) \leq m(n)$.

Definition. Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be defined by $m(n) = n + c$. A *cl-functional* is a m -bounded Turing functional.

We use Theorem 3.8 and Corollary 3.9 to derive the following data processing inequalities for sequences whose transformations are bounded Turing functionals.

Corollary 3.10. *If Φ is a cl-functional, then, for all $S, T \in \Sigma^\infty$ where Φ^S is defined,*

$$mdim(\Phi^S : T) \leq mdim(S : T)$$

and

$$Mdim(\Phi^S : T) \leq Mdim(S : T).$$

We also have a similar data processing inequality for h_α -bounded Turing functionals.

Corollary 3.11. *For all $\alpha \geq 1$, if Φ is a h_α -bounded Turing functional, then, for all $S, T \in \Sigma^\infty$ where Φ^S is defined,*

$$mdim(\Phi^S : T) \leq \alpha \cdot mdim(S : T)$$

and

$$Mdim(\Phi^S : T) \leq \alpha \cdot Mdim(S : T).$$

4 Turing Functionals with Bounded Yield and Reverse Data Processing Inequalities

In this section, we define the *yield* of a Turing functional and develop several reverse data processing inequalities (i.e., data processing inequalities where the transformations may significantly *increase* the mutual dimension between two sequences) using yield bounded Turing functionals.

We now introduce the *yield function* of a Turing functional.

Definition. The *yield function* of a Turing functional Φ equipped with oracle $S \in \Sigma^\infty$ is defined by

$$\phi_{yield}^S(n) = \min\{m \in \mathbb{N} \mid \Phi^{S \upharpoonright n}(m) \uparrow\},$$

for all $n \in \mathbb{N}$.

Intuitively, “use” is how much of the oracle the Turing functional must access in order for it to halt on a given input, and “yield” is how many inputs the Turing functional can halt on given a prefix of the oracle.

Definition. A sequence $T \in \Sigma^\infty$ is *yield bounded reducible* (*yb-reducible*) to a sequence $S \in \Sigma^\infty$ if T is Turing reducible to S by a Turing functional Φ such that ϕ_{yield}^S is bounded by a computable function.

For convenience, we say that T is *m-yb-reducible* to S if T is yb-reducible to S and $m : \mathbb{N} \rightarrow \mathbb{N}$ is a computable function bounding ϕ_{yield}^S .

In order to develop reverse data processing inequalities for sequences, we need to apply the following restriction to our Turing functionals.

Definition. A Turing functional Φ^S is *uniquely yielding* for an oracle $S \in \Sigma^\infty$ if, for all $T \in \Sigma^\infty$ and $n \in \mathbb{N}$,

$$\Phi^S \upharpoonright \phi_{yield}^S(n) \sqsubseteq \Phi^T \Rightarrow S \upharpoonright n \sqsubseteq T.$$

Definition. A sequence $T \in \Sigma^\infty$ is *uniquely yield bounded reducible* (*uyb-reducible*) to $S \in \Sigma^\infty$ if T is yb-reducible to S by a Turing functional that is uniquely yielding.

We say that T is *m-uyb-reducible* to S if T is uyb-reducible to S by a Turing functional whose yield function is bounded by a computable function $m : \mathbb{N} \rightarrow \mathbb{N}$.

Lemma 4.1. *If $T \in \Sigma^\infty$ is m-uyb-reducible to $S \in \Sigma^\infty$, then S is m-bT-reducible to T .*

Proof. Let T be m-uyb-reducible to S by a Turing functional Φ . We define a Turing functional Γ^T that operates on an input $n \in \mathbb{N}$ by querying the first $m(n)$ bits of T and searching for a string $x \in \Sigma^*$ such that $|x| \geq n$ and $\Phi^x(m(n)) = T \upharpoonright m(n)$. After finding x , Γ^T outputs $x \upharpoonright n$. Observe that

$$\begin{aligned} \Phi^S \upharpoonright \phi_{\text{yield}}^S(n) &\sqsubseteq \Phi^S \upharpoonright m(n) \\ &= T \upharpoonright m(n) \\ &= \Phi^x(m(n)) \\ &\sqsubseteq \Phi^x. \end{aligned}$$

Since Φ is uniquely yielding for S and $|x| \geq n$, $S \upharpoonright n \sqsubseteq x$, which implies that $\Gamma^T(n) = S \upharpoonright n$. \square

The following lemma follows directly by the Bounded Use Processing Lemma and Lemma 4.1.

Lemma 4.2 (Bounded Yield Processing Lemma). *Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be a increasing, computable function. For all $X, Y, Z \in \Sigma^\infty$, if Z is m-uyb-reducible to X , then*

$$mdim(X : Y) \leq mdim(Z : Y) \left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right)$$

and

$$Mdim(X : Y) \leq Mdim(Z : Y) \left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right),$$

except when $\left(\limsup_{n \rightarrow \infty} \frac{m(n)}{n} \right) = \infty$ while either $mdim(Z : Y) = 0$ or $Mdim(Z : Y) = 0$.

Definition. Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be defined by $m(n) = n + c$. A sequence $T \in \Sigma^\infty$ is *linear uniquely yield bounded reducible* (*ℓ -uyb-reducible*) to a sequence $S \in \Sigma^\infty$ if T is m-uyb-reducible to S .

The following theorem and corollary follow directly from the Bounded Yield Processing Lemma.

Theorem 4.3. *For all sequences $X, Y, Z \in \Sigma^\infty$, if Z is ℓ -uyb-reducible to X , then*

$$mdim(X : Y) \leq mdim(Z : Y)$$

and

$$Mdim(X : Y) \leq Mdim(Z : Y).$$

Corollary 4.4. *Let $\alpha \geq 1$. For all sequences $X, Y, Z \in \Sigma^\infty$, if Z is h_α -uyb-reducible to X , then*

$$mdim(X : Y) \leq \alpha \cdot mdim(Z : Y)$$

and

$$Mdim(X : Y) \leq \alpha \cdot Mdim(Z : Y).$$

The end of Section 3 discussed data processing inequalities in terms of the defined outputs of use bounded Turing functionals. In like manner, we describe reverse data processing inequalities in terms of yield bounded Turing functionals.

Definition. A Turing functional is a *yield bounded functional* (*yb-functional*) if there exists a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that, for every $S \in \Sigma^\infty$, $\phi_{yield}^S(n) \leq f(n)$.

Definition. A *uniquely yield bounded functional* (*uyb-functional*) is a yield bounded functional that is also uniquely yielding for every oracle.

For convenience, we say that a Turing functional is a *m-uyb-functional* if it is a uyb-functional whose yield is bounded by a computable function $m : \mathbb{N} \rightarrow \mathbb{N}$.

Definition. Let $m : \mathbb{N} \rightarrow \mathbb{N}$ be defined by $m(n) = n + c$. A Turing functional is a *linear uniquely yield bounded functional* (*ℓ-uyb-functional*) if it is a *m-uyb-functional*.

We use Theorem 4.3 and Corollary 4.4 to derive the following reverse data processing inequalities for sequences whose transformations are uniquely yield bounded Turing functionals.

Corollary 4.5. For all *ℓ-uyb-functionals* Φ and sequences $S, T \in \Sigma^\infty$ where Φ^S is defined,

$$mdim(S : T) \leq mdim(\Phi^S : T)$$

and

$$Mdim(S : T) \leq Mdim(\Phi^S : T).$$

Corollary 4.6. Let $\alpha \geq 1$. For all *h_α-uyb-functionals* Φ and sequences $S, T \in \Sigma^\infty$ where Φ^S is defined,

$$mdim(S : T) \leq \alpha \cdot mdim(\Phi^S : T)$$

and

$$Mdim(S : T) \leq \alpha \cdot Mdim(\Phi^S : T).$$

Acknowledgments

The author would like to thank Xiang Huang, Jack Lutz, Timothy McNicholl, and Donald Stull for useful discussions.

References

- [1] Klaus Ambos-Spies. Strongly Bounded Turing Reducibilities and Computably Enumerable Sets, 2011. Heidelberg University Lecture Notes.
- [2] C.B. Bell. Mutual information and maximal correlation as measures of dependence. *Annals of Mathematical Statistics*, 33(2):587–595, 1962.

- [3] Adam Case and Jack H. Lutz. Mutual dimension. *ACM Transactions on Computation Theory*, 7, July 2015.
- [4] Adam Case and Jack H. Lutz. Mutual dimension and random sequences. In *Proceedings of the 40th International Symposium on the Mathematical Foundations of Computer Science*, pages 199–210. Springer, 2015.
- [5] Thomas R. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., second edition, 2006.
- [6] Rod G. Downey, Denis R. Hirschfeldt, and Geoff LaForte. Randomness and reducibility. *Journal of Computer and System Sciences*, 68:96–114, 2004.
- [7] Péter Gács. Every sequence is reducible to a random one. *Information and Control*, 70:186–192, 1986.
- [8] Andrew E.M. Lewis and George Barmpalias. Random reals and lipschitz continuity. *Mathematical Structures in Computer Science*, 16:737–749, 2006.
- [9] Andrew E.M. Lewis and George Barmpalias. Randomness and the linear degrees of computability. *Annals of Pure and Applied Logic*, 145:252–257, 2006.
- [10] Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, third edition, 2008.
- [11] Jack H. Lutz. The dimensions of individual strings and sequences. *Information and Computation*, 187(1):49–79, 2003.
- [12] Elvira Mayordomo. A Kolmogorov complexity characterization of constructive Hausdorff dimension. *Information Processing Letters*, 84(1):1–3, 2002.
- [13] Hartley Rogers. *Theory of Recursive Functions and Effective Computability*. MIT Press, 1987.
- [14] Robert I. Soare. *Recursively Enumerable Sets and Degrees: A Study of Computable Functions and Computably Generated Sets*. Springer-Verlag, 1987.
- [15] Robert I. Soare. Turing oracle machines, online computing, and three displacements in computability theory. *Annals of Pure and Applied Logic*, 160:368–399, 2009.